UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/731,548	12/05/2003	Peter Szpak	42323-244143	4558

26694 7590 12/23/2008

VENABLE LLP
P.O. BOX 34385
WASHINGTON, DC 20043-9998

EXAMINER	
KE, PENG	
ART UNIT	PAPER NUMBER
2174	
MAIL DATE	DELIVERY MODE
12/23/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)	
	10/731,548	SZPAK ET AL.	
	Examiner	Art Unit	
	SIMON KE	2174	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *26 September 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3-16,18-25,27-33 and 37-41* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1, 3-16, 18-25, 27-33, and 37-41* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 10/08/08 has been entered.

Claims 1, 3-16, 18-25, 27-33, and 37-41 are pending in this application. Claims 1, 16, 30, 31, 32, and 33 are independent claims.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1, 3-5, 7-16, 18-20, 22-25, 27-33 and 37-41 are rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al. ("Coati" US Patent No. 6,851,105) in view of Little ("Little" US Patent 7,047,518) further in view of Liu US Patent 7,296,256.

Regarding independent claim 1, Coad teaches a method for generating embedded code from a graphical model (i.e. compare graphical model and generated embedded code in FIGS. 9-

12 et seq. of Coad), comprising the steps of: prompting a user to specify the embedded code from a plurality of embedded code (i.e. "Display pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad) and each embedded code relating to a characteristic of code to be generated from the graphical interface. (see Coad; col. 10 "table 1" teach pattern is related a characteristic of the code) Coad does not teach generating code for a code generation goal, the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; (see Little; col. 10, lines 25-55) Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;(see Little; col. 13, lines 40-55; The Class Diagrams and Perform analysis are done based on Main Use Case diagram which is also a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding dependent claim 3, Coad, Little, and Liu teach the method of claim 1, further comprising the step of providing feedback to the user regarding the compliance of the graphical model with a selected condition (i.e. "Are variations suitable" in FIG. 2 et seq. of Coad).

Regarding dependent claim 4, Coad, Little, and Liu teach the method of claim 3, wherein the user selects the selected condition through a user interface (i.e. see user selection of conditions in FIGS. 4-8 et seq. of Coad).

Regarding dependent claim 5, Coad, Little, and Liu teach the method of claim 4, wherein the user interface displays a list of conditions to be checked, and prompts the user to select one or more of the conditions (i.e. see user selection of conditions in FIGS. 4-8 et seq. of Coad).

Regarding dependent claim 7, Coad, Little, and Liu teach the method of claim 3, further comprising the step of modifying an object of the graphical model that does not comply with the

selected condition (i.e. compare steps 212-214 in FIG. 2 et seq. of Coad, defining special patterns for modifying objects).

Regarding dependent claim 8, Coad, Little, and Liu teach the method of claim 7, .wherein the step of modifying comprises identifying the object and prompting the user to manually modify a parameter of the object (see Little; col. 13, lines 40-55;).

Regarding dependent claim 9, Coad, Little, and Liu teach the method of claim 7, wherein the step of modifying comprises automatically modifying a parameter of the graphical model to comply with the selected condition (i.e. "Automatically ... to elements in pattern to pattern" in FIG. 15b et seq. of Coad).

Regarding dependent claim 10, Coad, Little, and Liu teach the method of claim 1, wherein the graphical model is a block diagram (see also FIG. 9-12 et seq. of Coad).

Regarding dependent claim 11, Coad in combination with Little teaches the method of claim 1, wherein each code generation goal corresponds to a general code generation goal (i.e. "pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad).

Regarding dependent claim 12, Coad in combination with Little teaches the method of claim 11, further comprising the step of prompting the user to specify at least one detailed code

generation goal for each specified general code generation goal (i.e. does special property need to be defined" in FIG. 2 et seq. of Coad).

Regarding dependent claim 13, Coad in combination with Little teaches the method of claim 12, further comprising the step of configuring the graphical model to comply with each detailed code generation goal (i.e. "pattern's configurable properties and parameters" in FIG. 2 et seq. of Coad).

Regarding independent claim 14, Coad teaches a method of preparing a graphical model for embedded code generation (i.e. compare graphical model and generated embedded code in FIGS. 9-12 et seq. of Coad), comprising the steps of: displaying a user interface for prompting a user and automatically changing parameters of the graphical model that are inconsistent (i.e. "Display pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad) and each embedded code relating to a characteristic of code to be generated from the graphical interface. (see Coad; col. 10 "table 1" teach pattern is related a characteristic of the code). Coad does not teach generating code for a code generation goal, the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) the at least one code generation goal being used

to generate embedded code from the graphical model in a graphical modeling environment; (see Little; col. 10, lines 25-55) Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;(see Little; col. 13, lines 40-55; The Class Diagrams and Perform analysis are done based on Main Use Case diagram which is also a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding dependent claim 15, Coad, Little, and Liu teach the method of, claim 14, further comprising the step of identifying a condition that does not comply with the code

generation goals specified by the user (i.e. "Check variations for ... errors" in FIG. 15b et seq. of Coad).

Regarding independent claim 16, Coad teaches a method of preparing a graphical model for embedded code generation (i.e. compare graphical model and generated embedded code in FIGS. 9-12 et seq. of Coad), the method comprising the steps of: displaying a graphical user interface though which a user can specify the embedded code to be generated from the graphical model (i.e. "Display pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad); providing feedback to the user regarding compliance of the graphical model (i.e. "Are variations suitable" in FIG. 2 et seq. of Coad) and each embedded code relating to a characteristic of code to be generated from the graphical interface. (see Coad; col. 10 "table 1" teach pattern is related a characteristic of the code). Coad does not teach generating code for a code generation goal, the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; (see Little; col. 10, lines 25-55) Changing parameters of the graphical model that are inconsistent with the at least one code generation goal; (see Little; col. 13, lines 40-55; The Class Diagrams

and Perform analysis are done based on Main Use Case diagram which is also a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding dependent claim 18, Coad, Little, and Liu teach the method of claim 16, acquiring at least one condition; and displaying information regarding compliance of the graphical model with the acquired at least one condition. (i.e. "Are variations suitable" in FIG. 2 et seq. of Coad).

Regarding dependent claim 19, Coad, Little, and Liu teach the method of claim 18, wherein the user selects the selected condition through the user interface (i.e. see user selection of conditions in FIGS. 4-8 et seq. of Coad).

Regarding dependent claim 20, Coad, Little, and Liu teach the method of claim 19, wherein the user interface displays a list of conditions to be checked, and prompts the user to select one or more of the conditions (i.e. see user selection of conditions in FIGS. 4-8 et seq. of Coad).

Regarding dependent claim 22, Coad, Little, and Liu teach the method of claim 18, further comprising the step of modifying an object of the graphical model that does not comply with the selected condition (i.e. compare stePS 212-214 in FIG. 2 et seq. of Coad, defining special patterns for modifying objects).

Regarding dependent claim 23, Coad, Little, and Liu teach the method of claim 22, wherein the step of modifying comprises identifying the object and prompting the user to manually modify a parameter of the object (see Little; col. 13, lines 40-55)

Regarding dependent claim 24, Coad, Little, and Liu teach the method of claim 22, wherein the step of modifying comprises automatically modifying a parameter of the graphical model to comply with the selected condition (i.e. "Automatically ... to elements in pattern to pattern" in FIG. 15b et seq. of Coad).

Regarding dependent claim 25, Coad, Little, and Liu teach the method of claim 16, wherein the graphical model is a block diagram (see FIG. 9-12 et seq. of Coad).

Regarding dependent claim 27, Coad, Little, and Liu teach the method of claim 16, wherein each code generation goal corresponds to a general code generation goal (i.e. "pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad).

Regarding dependent claim 28, Coad, Little, and Liu teach the method of claim 27, further comprising the step of prompting the user to specify at least one detailed code generation goal for each specified general code generation goal (i.e. does special property need to be defined" in FIG. 2 et seq. of Coad).

Regarding dependent claim 29, Coad, Little, and Liu teach the method of claim 28, further comprising the step of configuring the graphical model to comply with each detailed code generation goal (i.e. "pattern's configurable properties and parameters" in FIG. 2 et seq. of Coad).

Regarding independent claim 30, Coad teaches, in a graphical modeling environment, a medium holding computer-executable instructions for a method, comprising the steps of:

displaying a graphical user interface though which a user can specify one or more code to be generated from the graphical model (i.e. compare graphical model and generated embedded code in FIGS. 9-12 et seq. of Coad); and in response, providing feedback to the user regarding compliance of the graphical model (i.e. "Are variations suitable" in FIG. 2 et seq. of Coad) and each embedded code is relating to a characteristic of code to be generated from the graphical interface. (see Coad; col. 10 "table 1" teach pattern is related a characteristic of the code). Coad does not teach generating code for a code generation goal, the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; (see Little; col. 10, lines 25-55) Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;(see Little; col. 13, lines 40-55; The Class Diagrams and Perform analysis are done based on Main Use Case diagram which is also a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding independent claim 31, Coad teaches, in a graphical modeling environment, a medium holding computer- executable instructions for a method, comprising the steps of: displaying a user interface for prompting a user (i.e. "Display pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad); and automatically changing parameters of the graphical model that are inconsistent with the code generation goals specified by the user (i.e. "Automatically ... to elements in pattern to pattern" in FIG. 15b et seq. of Coad). Coad does not teach generating code for a code generation goal, the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; (see Little; col. 10, lines 25-55) Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;(see Little; col. 13, lines 40-55; The Class Diagrams and Perform analysis are done based on Main Use Case diagram which is also a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding independent claim 32, Coad teaches, in a graphical modeling environment, a medium holding computer- executable instructions for a method, comprising the steps of: prompting a user to specify for the embedded code (i.e. "Display pattern options corresponding to selected element type" in FIG. 2 et seq. of Coad) and each embedded code is relating to a characteristic of code to be generated from the graphical interface. (see Coad; col. 10 "table 1" teach pattern is related a characteristic of the code). Coad fail to teach generating code for a code generation goal, the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment; (see Little; col. 10, lines 25-55) Changing parameters of the graphical model that are inconsistent with the at least one code generation goal;(see Little; col. 13, lines 40-55; The Class Diagrams and Perform analysis are done based on Main Use Case diagram which is also a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding independent claim 33, Coad teaches an apparatus comprising: at least one processor; a memory coupled to the at least one processor; and a computer program residing in the memory and being executed by the at least one processor, wherein the computer program includes a wizard for guiding a user through a .process for preparing a graphical model for a code generation process for creating code based on the graphical model (i.e. "Receive a selected element type" and "Display pattern options" in FIG. 2 et seq. of Coad) and each embedded code is relating to a characteristic of code to be generated from the graphical interface. (see Coad; col. 10 "table 1" teach pattern is related a characteristic of the code). Coad does not teach at least one code generation goal specified by the user.

Little teaches generating code for a code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal)

It would have been obvious to an artisan at the time of the invention to include Little's teaching with method of Coad in order provide the user with a software code generator framework based on the user's design diagram goal.

However, they fail to teach the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation.

Liu teaches the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equation, the graphical model being capable of simulation based on the equation. (see Liu, col. 9, lines 35- col. 10, lines 57)

It would have been obvious to an artisan at the time of the invention to include Liu's teaching with method of Coad and Litter in order to provide user with an automating process-performance model that optimizes of IT infrastructure and application to reduce both equipment and labor cost.

Regarding dependent claim 37, Coad, Little, and Liu teach the apparatus of claim 33, wherein the wizard prompts the user to select one or more conditions to be checked in the graphical model (i.e. see user selection of conditions in FIGS. 4-8 et seq. of Coad).

Regarding dependent claim 38, Coad, Little, and Liu teach the apparatus of claim 37, wherein the wizard identifies objects in the graphical model that do not comply with the selected conditions (i.e. compare steps 212-214 in FIG. 2 et seq. of Coad, defining special patterns for

modifying objects).

Regarding dependent claim 39, Coad, Little, and Liu teach the apparatus of claim 37, wherein the wizard modifies objects in the graphical model that do not comply with the selected conditions (see Little; col. 14, lines 20-55).

Regarding dependent claim 40, Coad, Little, and Liu teach the method claim 1, Little further teaches where in at least code generation goal is a target application code generation goal, (see Little; col. 10, lines 25-55; The application model is a code generation goal) which is from the from group consisting of: a target application code generation goal; a maximum efficiency code generation goal; an aspect of memory code generation goal; an auto-generated identifier code generation goal; a comment code generation goal; an interface code generation goal; a model combination code generation goal; and a hypertext markup language report code generation goal.

Regarding dependent claim 41, Coad, Little and Liu teaches the method claim 1. Little teaches the at least one code goal is selected from the group consisting of:

A set of predefined options for an interface code generation goal comprising a passing data as arguments option and a not passing data as argument option; (see Little Fig. 11, col. 23, lines 25-47)

A set of predefined options for a target application code generation goal comprising a floating point target application option, a mixed point target application option, and a fixed point target application option;

a set of predefined options for a maximum efficiency code generation goal comprising a maximum efficiency option and a non-maximum efficiency option;

a set of predefined options for an aspect of memory code generation goal comprising a Random Access Memory (RAM) option and a Read-Only Memory (ROM) option;

a set of predefined options for an auto-generated identifier code generation goal comprising a verbose auto-generated identifier option and a non-verbose auto-generated identifier option;

a set of predefined options for a traceability code generation goal comprising a comments in the code option and a no comments in the code option;

a set of predefined options for a model combination code generation goal comprising a single model option and a multiple model option; and

a set of predefined options for a reporting code generation goal comprising an option for including a HTML report with the generated code and an option for not including a HTML report with the generated code.

Claims 6 and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Coad et al. ("Coati" US Patent No. 6,851,105) in view of Little ("Little" US Patent 7,047,518) in view of Liu US Patent 7,296,256 further in view of Frid-Nielsen ("Frid-Nielsen" US Patent No. 5,339,433)

Regarding dependent claim 6, Coad, Little, and Liu teach the method of claim 3, wherein the step of providing feedback to the user regarding the compliance of the graphical model with a selected condition comprises: Display non-comply with the selected condition (i.e. compare "Check variations for ... errors" in FIG. 15b et seq. of Coad with)

They fail to teach displaying a hyperlink for linking the selected condition to an object of the graphical model.

Frid-Nielsen teaches displaying a hyperlink for linking the selected condition to an object of the graphical model. (see. in FIG. 5 et seq. of Frid-Nielsen).

It would have been obvious to an artisan at the time of the invention to include Frid-Nielsen's teaching with method of Coad, Little, and Liu in order provide the user with dynamic access to object graphical model.

Regarding dependent claim 21, Coad, Little, and Liu teach the method of claim 18, wherein the step of indicating to the user whether the graphical model complies an object of the graphical model that does not comply with the selected condition (i.e. compare "Check variations for ... errors" in FIG. 15b et seq. of Coad with links in FIG. 5 et seq. of Frid-Nielsen).

They fail to teach displaying a hyperlink for linking the selected condition to an object of the graphical model.

Frid-Nielsen teaches displaying a hyperlink for linking the selected condition to an object of the graphical model. (see. in FIG. 5 et seq. of Frid-Nielsen).

It would have been obvious to an artisan at the time of the invention to include Frid-Nielsen's teaching with method of Coad, Little, and Liu in order provide the user with dynamic access to object graphical model.

Response to Arguments

Applicant's arguments with respect to claims 1, 3-16, 18-25, 27-33, and 37-41 have been considered but are moot in view of the new ground(s) of rejection.

Contact Information

Any inquiry concerning this communication or earlier communications from the examiner should be directed to SIMON KE whose telephone number is (571)272-4062. The examiner can normally be reached on M-Th and Alternate Fridays 8:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Stephen S. Hong can be reached on (571) 272-4124. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Peng Ke
/Peng Ke/
Primary Examiner, Art Unit 2174